Licence Sciences et Technologies
Université Joseph Fourier

Examen à mi-parcours
INF 232: Langages et Automates
L2, 2012/2013

Rappel des consignes et quelques conseils/remarques

– Durée : 2 heures (14h00 → 16h00).
– Aucune sortie avant 30 minutes. Aucune entrée après 30 minutes.
– Tout document du cours ou du TD est autorisé.
– Tout dispositif électronique est interdit (calculatrice, téléphone, tablette, etc.).
– Le soin de la copie sera pris en compte. Justifier soigneusement vos réponses.
– Les exercices sont indépendants.
– L'examen est sur 25 points. Le barème est donné à titre indicatif.

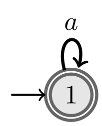
Solution de l'exercice 1

1. Faux. Les langages non-réguliers ne sont pas des langages d'états finis (théorème de Kleene). Pour ces langages, nous ne pouvons pas trouver d'automates d'états finis reconnaisseur. Par exemple, nous avons vu que le langage $\{a^n \cdot b^n \mid n \in \mathbb{N}\}$ n'est pas régulier.
2. Vrai. C'est le théorème de Kleene qui nous donne ce résultat : les langages réguliers sont (exactement) les langages d'états finis.
3. Vrai. En suivant la question 2, nous pouvons trouver un automate d'états finis reconnaisseur. Cet automate peut ensuite être minimisé. Un automate minimisé est tel quel tout automate avec un nombre strictement inférieur d'états reconnaît un langage différent.

Solution de l'exercice 2

Soit $w \in \{a, b\}^*$ un mot. L'exécution de w sur A suit une séquence de configurations. Sur A' l'exécution de w donnera la même séquence de configurations car les relations de transitions sont les mêmes. Ceci peut se démontrer par une induction sur la longueur de w.
Il faut observer également deux choses :
– L'alphabet Σ' de A' contient au moins tous les symboles de Σ utilisés par A dans sa relation de transition Δ. Autrement, A' ne serait pas vraiment bien défini (il aurait des symboles utilisés dans sa relation de transition qui ne figureraient pas dans son alphabet).
– Un mot accepté par A' ne contient aucun symbole dans $\Sigma' \setminus \Sigma$.
Nous en déduisons les résultats :

1. Vrai.
2. Vrai.
3. Faux. La question précédente donne déjà le résultat. Pour s'en convaincre un peu plus, prenons un contre-exemple :



avec $\Sigma = \{a\}$ et $\Sigma' = \{a, b\}$. Dans les deux cas, le langage accepté est a^*.

Solution de l'exercice 3

1. Nous allons utiliser deux algorithmes du cours :
– l'algorithme pour déterminer si le langage reconnu par un automate est vide ($\mathsf{est\_vide}$).
– la complémentation d'un automate ($\mathsf{complementation}$).
Nous avons donc $\mathsf{est\_universel}(ADEF \quad A) = \mathsf{est\_vide}\big(\mathsf{complementation}(A)\big)$.
2. La définition de déterminisme donne directement un algorithme. Un automate d'états-finis avec ensemble d'états Q, alphabet Σ, relation de transition Δ est déterministe si son alphabet ne contient

Algorithm 1 est_deterministe() pour déterminer si un automate est déterministe

Entrée : $A = (Q, \Sigma, \Delta, q_0, F)$ un automate (possiblement non-déterministe)
Sortie : `true` si l'automate est déterministe, `false` sinon

```
if ε ∈ Σ then
    return false
end if
for all state q ∈ Q do
    for all symbole s ∈ Σ do
        if |{q' ∈ Q | (q, s, q') ∈ Δ}| > 1 then
            return false ;
        end if
    end for
end for
return true ;
```

pas d'ϵ-transitions ($\epsilon \notin \Sigma$) et pour chaque état $q \in Q$, pour chaque symbole $s \in \Sigma$, le nombre de transitions dans Δ avec pour état de départ q et symbole s est au plus 1.

L'Algorithme 1 vérifie que ϵ n'appartient pas à l'alphabet de l'automate, puis réalise un parcours des états, puis un parcours de l'alphabet imbriqué. Pour chaque état et chaque symbole, l'algorithme vérifie qu'il y a *au plus* une transition définie à partir de l'état et sur un symbole donné.

On peut raffiner l'algorithme pour détailler le calcul de $\epsilon \in \Sigma$ et $|\{q' \in Q \mid (q, s, q') \in \Delta\}| > 1$.

- Pour déterminer si $\epsilon \in \Sigma$, on peut faire un parcours de Σ.
- Pour déterminer si $|\{q' \in Q \mid (q, s, q') \in \Delta\}| > 1$, dans le cas simple où Δ est représenté par une structure de données de type ensemble de triplets, on peut réaliser un parcours de Δ. Dans ce parcours, si les deux premiers éléments du triplet courant sont q et s respectivement, alors on incrémente un compteur. Ce compteur est initialisé à 0 pour chaque état et chaque symbole.

Nous obtenons l'Algorithme 2.

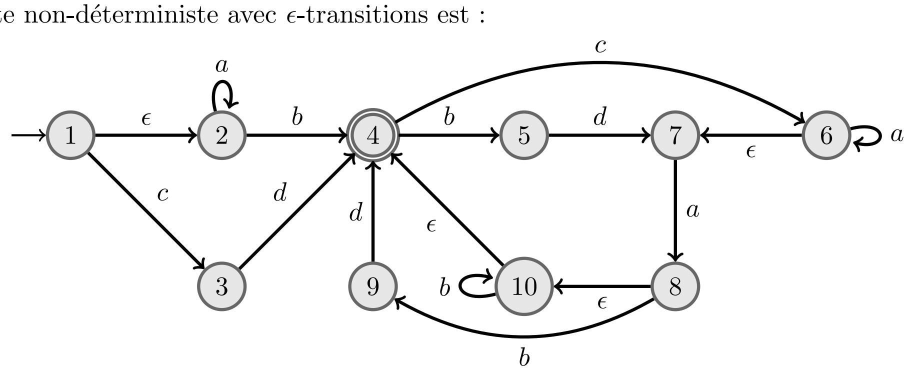
Solution de l'exercice 4

1. Nous observons d'abord que :
   - $a \cdot b \subseteq a^* \cdot b$, et
   - $a \cdot b^* + a \cdot b \cdot d = a(b^* + b \cdot d)$.

   L'expression régulière peut donc être légèrement simplifiée en :

$$\left(a^* \cdot b + c \cdot d\right) \cdot \left((c \cdot a^* + b \cdot d) \cdot a \cdot (b^* + b \cdot d)\right)^*$$

   L'automate non-déterministe avec ϵ-transitions est :

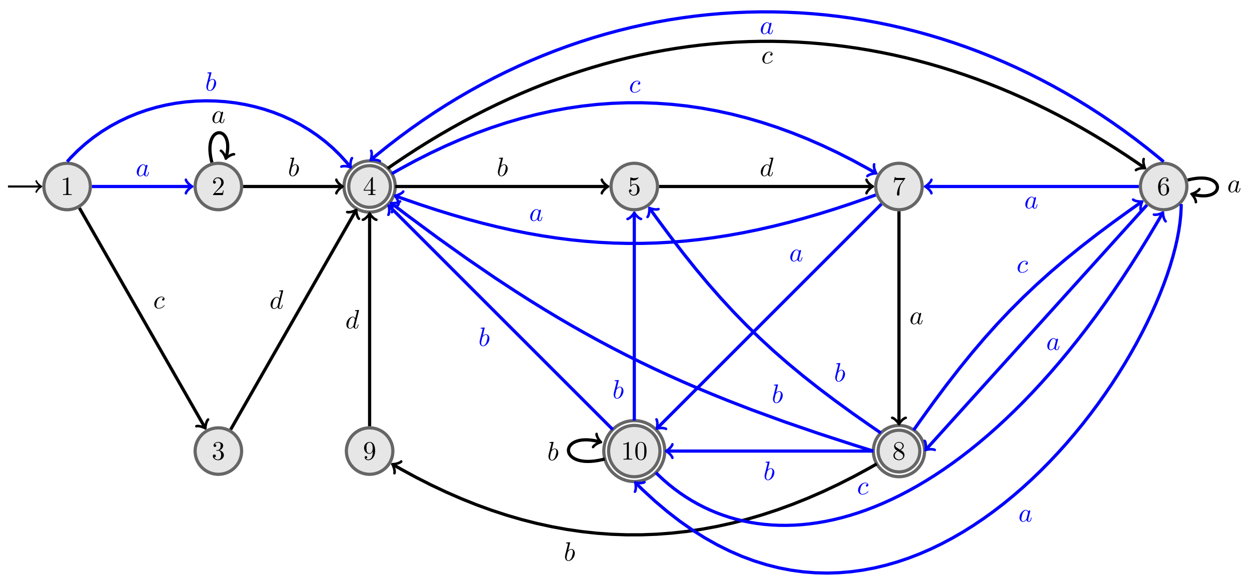


2. En éliminant les ϵ-transitions, nous obtenons :

Algorithm 2 est_deterministe() pour déterminer si un automate est déterministe - v2

Entrée : $A = (Q, \Sigma, \Delta, q_0, F)$ un automate (possiblement non-déterministe)
Sortie : `true` si l'automate est déterministe, `false` sinon

```
entier   compteur ;
for all symbole s ∈ Σ do
    if s = ε then
        return false ;
    end if
end for
for all state q ∈ Q do
    for all symbole s ∈ Σ do
        compteur := 0 ;
        for all (q1, sy, q2) ∈ Δ do
            if q1 = q ∧ sy = s then
                compteur := compteur + 1 ;
            end if
        end for
        if compteur > 1 then
            return false ;
        end if
    end for
end for
return true ;
```



3. Nous appliquons l'algorithme de déterminisation :

	$\{1\}$	$\{2\}$	$\{3\}$	$\{4\}*$	$\{5\}$	$\{6,7\}$	$\{7\}$	$\{4,6,7,8,10\}*$	$\{4,5,9,10\}*$	$\{4,7\}*$	$\{4,8,10\}*$	$\emptyset$
a	$\{2\}$	$\{2\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{4,6,7,8,10\}$	$\{4,8,10\}$	$\{4,6,7,8,10\}$	$\emptyset$	$\{4,8,10\}$	$\emptyset$	$\emptyset$
b	$\{4\}$	$\{4\}$	$\emptyset$	$\{5\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{4,5,9,10\}$	$\{5\}$	$\{5\}$	$\{4,5,9,10\}$	$\emptyset$
c	$\{3\}$	$\emptyset$	$\emptyset$	$\{6,7\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{6,7,10\}$	$\{6,7\}$	$\{6,7\}$	$\{6,7\}$	$\emptyset$
d	$\emptyset$	$\emptyset$	$\{4\}$	$\emptyset$	$\{7\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\{4,7\}$	$\emptyset$	$\emptyset$	$\emptyset$

L'état initial est $\{1\}$. Les états accepteurs sont marqués par une étoile : ce sont tous les états q tels que $q \cap \{4,8,10\} \neq \emptyset$, en d'autres termes, tous les états qui contiennent $4, 8$ ou 10.

4. Avant d'appliquer l'algorithme de minimisation, renommons les états. Nous obtenons :

	1	2	3	4*	5	6	7	8*	9*	10*	11*	12
a	2	2	12	12	12	8	11	8	12	11	12	12
b	4	4	12	5	12	12	12	9	5	5	9	12
c	3	12	12	6	12	12	12	6	6	6	6	12
d	12	12	4	12	7	12	12	12	10	12	12	12

L'algorithme de minimisation donne :

$\equiv_0$	$\equiv_1$	$\equiv_2$
1	1	1
2	2	2
3	3	3
5	5	5
6	6	6
7	7	7
12	12	12
4	4	4
8	8	8
9	9	9
10	10	10
11	11	11

L'algorithme termine après calcul de $\equiv_2$. Chaque état est dans une classe d'équivalence de cardinalité 1. L'automate est donc minimal.

Solution de l'exercice 5

1. Le système d'équations associé à cet automate est :

$$\begin{cases} X_1 = & aX_2 + bX_3 + cX_4 \\ X_2 = & aX_2 + bX_3 \\ X_3 = & aX_2 + bX_3 + \epsilon \\ X_4 = & dX_4 + \epsilon \end{cases}$$

À partir de $X_4 = dX_4 + \epsilon$, en appliquant le lemme d'Arden, comme $\epsilon \notin d$, nous obtenons $X_4 = d^*$. À partir de $X_2 = aX_2 + bX_3$, en appliquant le lemme d'Arden, comme $\epsilon \notin a$, nous obtenons $X_2 == a^*(bX_3) = a^*bX_3$.
En substituant dans $X_3 = aX_2 + bX_3 + \epsilon$, nous obtenons :

$$\begin{cases} X_1 = & aX_2 + bX_3 + cX_4 \\ X_2 = & a^*bX_3 \\ X_3 = & a^*bX_3 + bX_3 + \epsilon \\ X_4 = & d^* \end{cases}$$

Comme $bX_3 \subseteq a^*bX_3$, nous obtenons :

$$\begin{cases} X_1 = & aX_2 + bX_3 + cX_4 \\ X_2 = & a^*bX_3 \\ X_3 = & a^*bX_3 + \epsilon \\ X_4 = & d^* \end{cases}$$

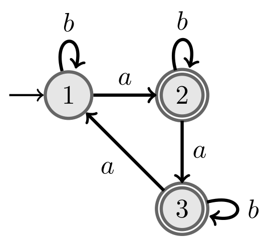
À partir de $X_3 = a^*bX_3 + \epsilon$, en appliquant le lemme d'Arden, comme $\epsilon \notin a^*b$, nous obtenons $X_3 = (a^*b)^*$.
En substituant dans $X_2 = a^*bX_3$, nous obtenons :

$$\begin{cases} X_1 = & aX_2 + bX_3 + cX_4 \\ X_2 = & a^*b(a^*b)^* \\ X_3 = & (a^*b)^* \\ X_4 = & d^* \end{cases}$$

Finalement $X_1 = a^+ \cdot b(a^* \cdot b)^* + b \cdot (a^* \cdot b)^* + c \cdot d^*$.

Solution de l'exercice 6

1. D'abord, nous renommons les états de l'automate.



Nous suivons la méthode vue en cours, les équations du cas de base sont :

$$\begin{array}{lll} R^0_{11} = b + \epsilon & R^0_{21} = \emptyset & R^0_{31} = a \\ R^0_{12} = a & R^0_{22} = b + \epsilon & R^0_{32} = \emptyset \\ R^0_{13} = \emptyset & R^0_{23} = a & R^0_{33} = b + \epsilon \end{array}$$

L'équation inductive est :

$$R^k_{ik} = R^{k-1}_{ij} + R^{k-1}_{ik} \cdot (R^{k-1}_{kk})^* \cdot R^{k-1}_{kj}$$

Nous faisons donc le calcul en partant de $k = 1$ jusqu'à $k = 3$ (car le plus grand état est numéroté 3).
Pour $k = 1$, nous avons les équations suivantes :

$$\begin{array}{lll} R^1_{11} = (b+\epsilon) + (b+\epsilon)(b+\epsilon)^*(b+\epsilon) = b^* & R^1_{21} = \emptyset & R^1_{31} = a + a(b+\epsilon)^*\emptyset = a \\ R^1_{12} = a + (b+\epsilon)(b+\epsilon)^*a = b^*a & R^1_{22} = b + \epsilon & R^1_{32} = a(b+\epsilon)^*a = ab^*a \\ R^1_{13} = (b+\epsilon)(b+\epsilon)^*\emptyset = \emptyset & R^1_{23} = a & R^1_{33} = b + \epsilon + a(b+\epsilon)^*\emptyset = b + \epsilon \end{array}$$

Pour $k = 2$, nous avons les équations suivantes :

$$\begin{array}{ll} R^2_{11} = b^* + b^*a(b+\epsilon)^*\emptyset = b^* & R^2_{21} = (b+\epsilon)(b+\epsilon)^*\emptyset = \emptyset \\ R^2_{12} = b^*a + b^*a(b+\epsilon)^*(b+\epsilon) = b^*ab^* & R^2_{22} = (b+\epsilon) + (b+\epsilon)(b+\epsilon)^*(b+\epsilon) = b^* \\ R^2_{13} = b^*a(b+\epsilon)^*a = b^*ab^*a & R^2_{23} = a + (b+\epsilon)(b+\epsilon)^*a = b^*a \end{array}$$

$$\begin{array}{l} R^2_{31} = a + ab^*a(b+\epsilon)^*\emptyset \\ R^2_{32} = ab^*a + ab^*a(b+\epsilon)^*(b+\epsilon) = ab^*ab^* \\ R^2_{33} = (b+\epsilon) + ab^*a(b+\epsilon)^*a = \epsilon + b + ab^*ab^*a \end{array}$$

Pour $k = 3$, il nous suffit de calculer R^3_{12} et R^3_{13} car l'état initial est 1 et les deux états accepteurs sont 2 et 3 :

$$\begin{aligned} R^3_{12} &= R^2_{12} + R^2_{13}(R^2_{33})^* R^2_{32} \\ &= b^*ab^* + b^*ab^*a(\epsilon + b + ab^*ab^*a)^* ab^*ab^* \end{aligned}$$

$$\begin{aligned} R^3_{13} &= R^2_{13} + R^2_{13}(R^2_{33})^* R^2_{33} \\ &= R^2_{13} + R^2_{13}(R^2_{33})^+ \\ &= R^2_{13}(R^2_{33})^* \\ &= b^*ab^*a(\epsilon + b + ab^*ab^*a)^* \\ &= b^*ab^*a(b + ab^*ab^*a)^* \end{aligned}$$

Au final, l'expression régulière associée à cet automate est :

$$R^3_{12} + R^3_{13}$$

Remarque 1. Pour rappel, cette méthode peut être utilisée pour démontrer le théorème de Kleene (la moitié). Mais c'est lourd et long. Il faut calculer n^3 expressions régulières pour un automate à n états. Heureusement ici n vaut 3.

Remarque 2. Juste à titre de remarque (ceci ne constituait pas une réponse à la question). Comme l'automate est facile on peut aussi utiliser une méthode plus intuitive et un peu dans le même esprit. Entre deux passages sur l'état 1 consécutifs il y a 3 possibilités :

1. soit ϵ,
2. soit b,
3. soit ab^*ab^*a.

Maintenant

$R(L_{11}) =$ $(b + ab^*ab^*a)^*$ // ici dernière visite en 1
// puis on termine en atteignant un état accepteur sans revenir sur 1 :
$(ab^* + ab^*ab^*)$

d'où le résultat :

$$(b + a \cdot b^*a \cdot b^* \cdot a)^* \cdot (a \cdot b^* + a \cdot b^* \cdot a \cdot b^*)$$

Solution de l'exercice 7

1. Notons L le langage de l'énoncé. Trois preuves sont possibles (au moins) :
   - On peut remarquer que $\{a^m \cdot b^n \cdot c^p \mid m \geq 0 \text{ et } n > p \geq 0\} \cap L(b^* \cdot c^*) = \{b^i \cdot c^j \mid i > j\}$. D'après la fermeture des langages réguliers, si $\{a^m \cdot b^n \cdot c^p \mid m \geq 0 \text{ et } n > p \geq 0\}$, alors son intersection avec $L(b^* \cdot c^*)$ qui est régulier, donnerait un langage régulier. Nous avons montré dans le cours que $\{b^i \cdot c^j \mid i > j\}$ n'est pas régulier.
   - Considérons l'application $h : \{a, b, c\} \to \{x, y\}^*$ telle que $h(a) = h(b) = x$ et $h(c) = y$. Cette application induit un homomorphisme de $\{a, b, c\}^*$ vers $\{x, y\}^*$. Nous avons

     $$\begin{aligned} h(L) = &\ \{x^m \cdot x^n \cdot y^p \mid n > p \text{ et } m \geq 0\} \\ = &\ \{x^{m+n} \cdot y^p \mid n > p \text{ et } m \geq 0\} \\ = &\ \{x^n \cdot y^p \mid n > p\} \end{aligned}$$

     La dernière égalité se montrant assez facilement par inclusion des ensembles l'un dans l'autre. Dans le cours, nous avons montré que $\{x^n y^p \mid n > p\} = h(L)$ n'était pas régulier. En utilisant la fermeture des langages réguliers par homomorphisme, nous obtenons par contraposition que L n'est pas régulier.

– Une démonstration complète par contradiction est aussi possible. Supposons que L soit régulier. D'après le lemme de l'itération, il existe une constante $n \in \mathbb{N}$ telle que, pour tout mot w de L de longueur supérieure à n, il existe une décomposition de w avec trois sous-mots $x, y, z \in \{a, b, c\}^*$ tels que :
  – $w = x \cdot y \cdot z$
  – $|x \cdot y| \leq n$
  – $y \neq \epsilon$
  – $\forall k \in \mathbb{N} : x \cdot y^k \cdot z \in L$.

  Considérons le mot $w = b^{n+1} \cdot c^n$. Nous avons que $w \in L$ $(m = 0)$ et $|w| \geq n$. En utilisant $w = xyz$, $|xy| \leq n$, on obtient $y = b^i$ avec $i \geq 0$. En utilisant $y \neq \epsilon$, nous obtenons $i > 0$. En utilisant $\forall k \in \mathbb{N} : xy^k z \in L$, pour $k = 0$, nous obtenons que $b^{n+1-i}c^n \in L$. Or $n + 1 - i \leq n$ car $i \geq 1$. Ceci est une contradiction.